Self-attention-based Diffusion Model for Time-series Imputation

Mohammad Rafid Ul Islam[1], **Prasad Tadepalli**[1], **Alan Fern**[1]

[1]Oregon State University
islammoh@oregonstate.edu, prasad.tadepalli@oregonstate.edu, Alan.Fern@oregonstate.edu

Abstract

Time-series modeling is essential for applications in agriculture, weather forecasting, food production, and more. However, missing data due to sensor malfunctions, power outages, and human errors is a common issue, complicating the training of machine learning models. We propose a diffusion-based generative model to address this problem and fill the gaps in the data. Our approach captures feature and time correlations through a two-stage imputation process. Our model outperforms state-of-the-art imputation methods and is more scalable in GPU resources.

1 Introduction

Many applications in agriculture and atmospheric science rely heavily on time-series data. Time-series data modeling is essential for accurate weather forecasting, predicting agricultural yields, and understanding environmental patterns. For instance, in agriculture, these models help predict crop growth, optimize irrigation schedules, and forecast food production, enabling better resource management and planning. In meteorology and climate science, time-series models are used to predict weather conditions, monitor climate change, and prepare for extreme weather events.

However, one significant challenge in time-series data modeling in agricutural applications is the presence of missing data. Missing data can result from various factors, including sensor malfunctions, power outages, and human errors (Silva et al. 2012; Yi et al. 2016). Missing data prevents users, including farmers, from making well-informed decisions, as the incomplete data hampers their ability to make accurate assessments and predictions. Furthermore, these gaps in data complicate the training of machine learning models, leading to less accurate predictions and reduced reliability of the models. It is, therefore, critical to address the issue of missing data to improve the performance and robustness of time-series models.

Imputation is a method used to fill in missing values in a dataset. In the context of time-series data, imputation techniques help estimate and replace missing values based on observed data. This enables the continuation of accurate and reliable data analysis. In agricultural and weather-related applications, imputation is crucial as it helps maintain the integrity of the dataset. This, in turn, leads to better predictions, improved resource management, and more informed decision-making.

Time-series data imputation has been studied in both statistical learning and deep learning communities. Traditional statistical learning techniques include mean/median imputation, linear interpolation, K-nearest neighbor imputation, and the more advanced iterative regression-based models such as MICE (van Buuren and Groothuis-Oudshoorn 2011). With advancements in deep learning, neural network-based models such as BRITS (Cao et al. 2018), SAITS (Du, Côté, and Liu 2023), GRUI-GAN (Luo et al. 2018), and GP-VAE (Fortuin et al. 2020) have been developed for time-series imputation. There have been some GAN-based models like (Luo et al. 2019, 2018; Liu et al. 2019; Miao et al. 2021) and VAE-based methods like (Fortuin et al. 2020). However many of these methods suffer from unstable training. Score-based generative modelling has achieved substantial performance in recent years. Diffusion models like CSDI (Tashiro et al. 2021) and SSSD (Alcaraz and Strodthoff 2022) provide high-quality imputations by conditioning on observed values. Another notable diffusion model, TimeGrad (Rasul et al. 2021), excels in forecasting but cannot leverage future data for imputation. Other effective methods include Bayesian inference models (Cui et al. 2019; Vidotto, Vermunt, and Van Deun 2018, 2019), graph neural network-based solution GRIN (Cini, Marisca, and Alippi 2021), latent ODE networks with RNNs (Rubanova, Chen, and Duvenaud 2019), and Schrodinger-bridge method (Chen et al. 2023). Compared to other models, diffusion models have the advantage that they naturally generate a distribution of possible completions, which captures the inherent uncertainty in the data.

In this paper, we propose a diffusion-based generative model, **S**elf **A**ttention-based **D**iffusion Model for Time Series **I**mputation (SADI), to fill in the gaps in time-series data and evaluate it on weather-related datasets. Our approach models both feature and temporal correlations through self-attention mechanisms and employs a two-stage imputation process, ensuring that the imputed values are realistic and consistent with the observed data. This method not only enhances the accuracy of the predictions but also scales ef-

ficiently with GPU resources, making it suitable for large-scale applications. Our experiments demonstrate that our model outperforms state-of-the-art imputation methods, offering a reliable solution for handling missing data in time-series datasets. In summary, our contributions are as follows.

1. We introduce a novel diffusion-based generative model for time-series imputation.
2. Our model captures the feature and temporal correlations with two distinct components: *feature dependency encoder* (FDE) and *gated temporal attention* (GTA). In addition, we have a two-stage imputation process where the second stage improves the imputation predictions from the first stage.
3. We evaluate our model in several agriculture and weather-related time-series datasets and show that it outperforms the state-of-the-art.

2 Background: Diffusion Models

We represent mutivariate time-series data as $X = \{x_{1:L,1:K}\} \in \mathbf{R}^{L \times K}$, where K is the number of features and L is the time-series length. Let $\mathbf{P}(X)$ be the original joint distribution of the time-series data. We observe only part of it, X_{obs}, while X_{miss} represents the missing parts. We use a binary mask, $M = \{m_{1:L,1:K}\} \in \{0,1\}^{L \times K}$, to keep track of the missing data. Here, 0 indicates missing data and 1 indicates observed data. We generate missing data according to $\mathbf{P}(X_{miss}|X_{obs})$.

We employ a diffusion-based generative model to impute the missing values. A diffusion model has two processes: a forward process and a reverse process. The forward method gradually introduces noise into the initial data until it degenerates into pure noise. The reverse technique then eliminates the noise little by little, beginning with pure noise and tries to reconstruct the original data distribution in the same number of steps (Sohl-Dickstein et al. 2015; Ho, Jain, and Abbeel 2020).

The idea behind the diffusion model is to estimate the data distribution $q(X_0)$ by training a model distribution $p_\theta(X_0)$. The latent variables X_t for represent the noisy data at diffusion step $t \in \{1, \ldots, T\}$ and belong to the same sample space as the original data, X_0. The forward diffusion process forms a Markov chain and is defined by

$$q(X_{1:T}|X_0) = \prod_{t=1}^{T} q(X_t|X_{t-1})$$

$$q(X_t|X_{t-1}) = \mathcal{N}(\sqrt{1-\beta_t}X_{t-1}, \beta_t\mathbf{I}) \quad (1)$$

In the given context, β_t denotes the variance of the noise added at each step t of the forward process. Additionally, the expression for X_t has a closed form as $X_t = \sqrt{\bar{\alpha}_t}X_0 + \sqrt{(1-\bar{\alpha}_t)}\epsilon$, where $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$, and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The reverse process, modelled by p_θ, is also a Markov chain that uses a denoising function ϵ_θ to denoise X_t and obtain X_{t-1}.

$$p_\theta(X_{0:T}) = p(X_T)\prod_{t=1}^{T} p_\theta(X_{t-1}|X_t), \text{ where } X_T \sim \mathcal{N}(0, \mathbf{I})$$

$$p_\theta(X_{t-1}|X_t) = \mathcal{N}(\mu_\theta(X_t, t), \sigma_\theta(X_t, t)\mathbf{I}) \quad (2)$$

Now, following (Ho, Jain, and Abbeel 2020), we have:

$$\mu_\theta(X_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(X_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(X_t, t)\right) \quad (3)$$

$$\sigma_\theta(X_t, t) = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t \quad (4)$$

In Eq. 3, the denoising function ϵ_θ is a neural network. Using the parameterization of $\mu_\theta(X_t, t)$ in Eq. 3, (Ho, Jain, and Abbeel 2020) have shown that the reverse process can be trained by optimizing the following objective:

$$L = \min_\theta \mathbb{E}_{X \sim q(X_0), \epsilon \sim \mathcal{N}(0,\mathbf{I}), t} ||\epsilon - \epsilon_\theta(X_t, t)||_2^2 \quad (5)$$

Once the training is completed, we can sample X_0 by following the expressions outlined in Eqs. (2) and (3).

3 Self-attention-based Diffusion Model for Time-series Imputation

Our model, SADI, uses a conditional diffusion process, where the model conditions on the observed data, X_0^{co}. Figure 1 shows the architecture of the neural-network-based denoising model, $\epsilon_\theta(X_t^{ta}, X_0^{co}, t)$, following Eq. 3. Here, X_t^{ta} represents the noisy data at diffusion step t. The denoising model has three main features: (1) a *feature dependency encoder* (FDE), which is responsible for modeling feature correlations; (2) a *gated temporal attention* block (GTA), which captures the temporal dependencies; and (3) a *two-stage imputation* process which refines and improves the initial imputation.

The diffusion denoising function is utilized to estimate the noise introduced by the forward process. However, as indicated by Eq. 3, it's clear that predicting the noise is equivalent to predicting the imputation. Therefore, we revolve the following discussion around predicting the imputation instead of noise.

Feature Dependency Encoder

The *feature dependency encoder* (FDE) has two components: a 1-D dilated convolution (kernel size 1×3) and a self-attention mechanism with layer normalization and a feed forward network. The dilated convolution captures the local relationships in the temporal dimension. In every FDE layer, we increment the dilation by 1 to extend the receptive field and capture more extensive local information. Then, we use the attention mechanism putting attention on the feature dimension. This makes the model learn the joint-time-series level relationships leading to better imputations.

The FDE processes the original values and the noisy data $(X_0^{co} + X_t^{ta})$ of dimension (L, K), while taking into account the missingness mask M_0^{co} of the same dimensions. Additionally, categorical positional encoding is utilized to differentiate between various features within the feature dimension. The number of FDE layers is controlled by a hyperparameter N_{FDE}. The operations for FDE are illustrated in Eq. 6. The resulting representation $\hat{X}$ has a size of (L, K).

$$X = (feature\_pos\_enc(concat((X_0^{co} + X_t^{ta}), M_0^{co})))^T$$

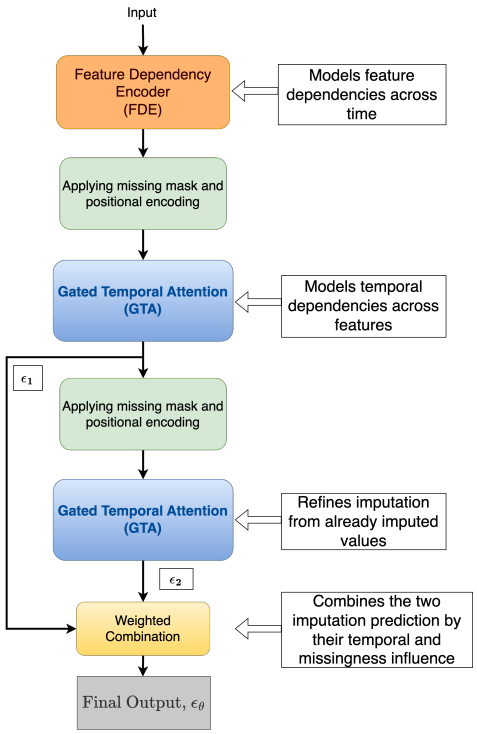


Figure 1: Overview of the architecture of our neural network model that predicts the denoising function, ϵ_θ

$$\hat{X} = \begin{cases} FDE_n(X) & \text{if } n = 1 \\ FDE_n(\hat{X}) & \text{if } 1 < n \leq N_{FDE} \end{cases} \quad (6)$$

Gated Temporal Attention

The *gated temporal attention* block puts the attention on the time dimension to capture the temporal dependencies. The GTA follows the residual block architecture of DiffWave (Kong et al. 2020) and WaveNet (Oord et al. 2016) models. However, we use two back-to-back self-attention layers focusing attention on the time dimension instead of using the dilated convolutions from WaveNet to capture the temporal correlations. We also apply positional encoding on the time dimension to indicate it as a sequence for the self-attention mechanism. The GTA block has a gated linear unit (GLU) activation function applied to the outputs of the self-attention layers, which is the reason for naming it the *gated temporal attention*.

There are multiple layers of GTA in a GTA block. The hyperparameter N_{GTA} controls the number of layers within the GTA block. The block takes the diffusion step embedding, t_{emb}, the output from FDE, denoted as $\hat{X}$ from Eq. 6, and the conditional observed data X_0^{co} as inputs, after applying the missing mask M_0^{co} and a positional encoding to the inputs. Then, we project X_0^{co} and $\hat{X}_1$ from dimensions (L, K) to dimensions (L, D). The GTA block takes the projected version of the inputs and applies the attention mechanism. The GTA block produces three outputs: a hidden state $\tilde{X}$ (L, D) which is used as input for the subsequent GTA layer, a skip connection $\epsilon'(L, D)$ contributing to the intermediary imputation, and attention weights $W_L(L, L)$. To get the interim imputation ϵ_1, we sum over all ϵ' skip connections and project them back to (L, K) dimensions according to Eq. 10.

$$\hat{X}_{pos_1} = time\_pos\_enc(linear(concat(\hat{X}_1, M_0^{co}))) \quad (7)$$

$$X_{pos}^{co} = time\_pos\_enc(linear(concat(X_0^{co}, M_0^{co}))) \quad (8)$$

$$\tilde{X}, W_L, \epsilon'_n = \begin{cases} GTA_n^i(\hat{X}_{pos_i}, X_{pos}^{co}, t_{emb}) & \text{if } n = 1 \\ GTA_n^i(\tilde{X}, X_{pos}^{co}, t_{emb}) & \text{if } 1 < n \leq N_{GTA} \end{cases} \quad (9)$$

$$\epsilon_1 = linear\left(\frac{\sum_{n=1}^{N_{GTA}} \epsilon'_n}{\sqrt{2}}\right) \quad (10)$$

Two-stage Imputation Process

After passing through multiple FDE and GTA layers, the data is transformed to handle feature and temporal dependencies. To preserve original data characteristics and enhance imputation quality, we reintegrate the original noisy data. This grounding step ensures the *second GTA block* utilizes both transformed and original data for improved results.

In the first stage, each GTA layer passes the hidden state $\tilde{X}$ to the next GTA layer without directly receiving imputation information (ϵ'), which is aggregated into the interim imputation ϵ_1 at the end of the first block. In the second stage, ϵ_1 is incorporated into the input for GTA operations to improve missing value prediction. Initially, there are no imputed values to assist, but once generated, they guide subsequent imputations, capturing relationships between observed and imputed data and their impact on other missing data points.

The second GTA block performs the same operations as described in Section 3, resulting in another interim imputation, ϵ_2, and an attention weight matrix, W_L, of dimension (L, L). We introduce the original noisy data X_t^{ta} into the input of the second GTA to obtain the second interim imputation ϵ_2.

$$\hat{X}_2 = \tilde{X} + \epsilon_1 + X_t^{ta}$$
$$\hat{X}_{pos_2} = time\_pos\_enc(linear(concat(\hat{X}_2, M_0^{co}))) \quad (11)$$

$$\tilde{X}, W_L, \epsilon'_n = \begin{cases} GTA_n^2(\hat{X}_{pos_2}, X_{pos}^{co}, t_{emb}) & \text{if } n = 1 \\ GTA_n^2(\tilde{X}, X_{pos}^{co}, t_{emb}) & \text{if } 1 < n \leq N_{GTA} \end{cases} \quad (12)$$

$$\epsilon_2 = linear\left(\frac{\sum_{n=1}^{N_{GTA}} \epsilon'_n}{\sqrt{2}}\right) \quad (13)$$

We combine ϵ_1 and ϵ_2 to get the final imputation, ϵ_θ, as illustrated in Eq. 15. The weights $\hat{W}_L$ decide how much of

Algorithm 1: Training of our diffusion model

Input: Distribution of training data $X_0 \sim q(X_0)$, the number of iteration/epochs E, the list of noise levels $(\bar{\alpha}_1, \ldots, \bar{\alpha}_T)$
Output: Trained ϵ_θ denoising function

1: **for** $i = 0$ to E **do**
2: $t \sim Uniform(\{1, \ldots, T\})$
3: Separate X_0 into conditional observations X_0^{co} and imputation targets X_0^{ta}
4: Noise $\epsilon = \mathcal{N}(0, \mathbf{I})$ with the same dimension as X_0^{ta}
5: One step calculation to noisy targets at step t, $X_t^{ta} = \sqrt{\bar{\alpha}_t} X_0^{ta} + \sqrt{(1 - \bar{\alpha}_t)}\epsilon$
6: denoising function prediction, $\epsilon_1, \epsilon_2, \epsilon_\theta = \epsilon_\theta(X_t^{ta}, X_0^{co}, t)$
7: Optimize the loss function for ϵ_θ, ϵ_1, and ϵ_2 according to Eq. (5) and (16).
8: **end for**

each interim imputation should contribute to the final imputation. We get the weights from the second stage's attention weights, W_L, after applying the missing mask, M_0^{co}, and projecting the result to (L, K) dimensions as shown in Eq. 14.

$$\tilde{W}_L = sigmoid(linear(concat(W_L, M_0^{co}))) \tag{14}$$

$$\epsilon_\theta = (1 - \tilde{W_L}) \odot \epsilon_1 + \tilde{W_L} \odot \epsilon_2 \tag{15}$$

Here, $\odot$ refers to the element-wise product between two matrices/tensors.

Training and Sampling

We followed the training infrastructure of DDPM (Ho, Jain, and Abbeel 2020), which is shown in Algorithm 1. In each training epoch, a diffusion step t is sampled uniformly from the set $\{1, 2, \ldots, T\}$. Then, we create the imputation target X_0^{ta} by randomly omitting a percentage of the data to create the ground truth for optimization. We use the closed form of the forward diffusion process in Step 5. Then, we get the three noise predictions ϵ_1, ϵ_2, and ϵ_θ (the two intermediary imputations and the final one) from our denoising model. Finally, we optimize the three noise predictions for the imputation targets by following Eq. 5 and Eq. 16.

$$loss = \frac{M_0^{ta}}{2N} \left(||\epsilon - \epsilon_\theta||_2^2 + \frac{(||\epsilon - \epsilon_1||_2^2 + ||\epsilon - \epsilon_2||_2^2)}{2} \right) \tag{16}$$

Here, N is the number of imputation targets, and M_0^{ta} is the target mask where 1 indicates the targets for imputation and 0 represents observed values and original missing data (without ground truth).

The sampling procedure follows the reverse diffusion process. This is an iterative process as shown in Algorithm 2. This procedure starts with a pure noise $X_T^{ta} \sim \mathcal{N}(0, \mathbf{I})$ in place of the missing data and gradually removes the noise to

Algorithm 2: Sampling process

Input: Data sample X_0, missingness mask M_0^{co}, total number of diffusion steps T, trained denoising function ϵ_θ
Output Imputed missing values X_0^{ta}

1: $X_0^{co} =$ observed values of X_0
2: $X_{curr} = X_T^{ta} \sim \mathcal{N}(0, \mathbf{I})$ (same dimensions as X_0)
3: **for** $t = T$ to 1 **do**
4: $\epsilon_\theta = \epsilon_\theta(X_{curr}, X_0^{co}, M_0^{co})$
5: $\mu_\theta = \frac{1}{\sqrt{\alpha_t}}(X_{curr} - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta)$
6: $\sigma_\theta = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$;
7: **if** $t = 0$ **then**
8: $X_{curr} = \mathcal{N}(\mu_\theta, \mathbf{I})$
9: **else**
10: $X_{curr} = \mathcal{N}(\mu_\theta, \sigma_\theta \mathbf{I})$
11: **end if**
12: **end for**
13: $X_0^{ta} = X_{curr}$
14: $X_0 = X_0^{co} \times M_0^{co} + X_0^{ta} \times (1 - M_0^{co})$

get the imputed data. In each step of the sampling, we predict the noise ϵ_θ at step t and create the mean (μ_θ) and the variance (σ_θ) of step $t - 1$ according to Step 5 and Step 6 of Algorithm 2. Step 14 shows the calculation of the final output (X_0) from the observed data (X_0^{co}) and imputed data (X_0^{ta}). We generate 100 such samples and take the mean of that as our final imputation.

4 Experiments

We use three agricultural and weather datasets for the evaluation of our model. The following discussion introduces the datasets and describes the results.

Datasets

The **AgAID** dataset contains grape cultivar cold hardiness data and some plant features along with some environmental features (Institue 2023). The plant-related features were collected by the viticulture team from Washington State University,[1] and the environmental sensor data was acquired through the AgWeatherNet API.[2] The dataset contains dormant seasons from September 7 to May 15, with 21 features and 252 time steps (days). It has a total of 34 seasons (1988 to 2022). The last two seasons are taken as test data.

The **NACSE** (National Alliance for Computational Science & Engineering) PRISM climate data (PRISM 2014) has the maximum and minimum temperatures recorded daily across 176 weather stations in Western Oregon, which adds up to 352 features and 366 time steps (days). The dataset contains 11 years worth of data from January 1, 2011 to December 31, 2021. In our setup, we take the first 9 years as training and the last 2 as test data.

The **Air Quality** dataset is a popular benchmarking dataset for time-series imputation (Yi et al. 2016). We use

[1] https://wine.wsu.edu/
[2] https://weather.wsu.edu/

Dataset	Model	Random missing (%)		
		20%	50%	80%
AgAID	MICE	4.26e-02 ± 1.23e-03	5.91e-02 ± 6.64e-04	6.77e-02 ± 4.40e-04
	BRITS	2.76e-02 ± 7.34e-04	3.61e-02 ± 7.70e-04	4.89e-02 ± 6.79e-04
	SAITS	1.22e-02 ± 7.41e-04	1.97e-02 ± 5.22e-04	4.22e-02 ± 8.57e-04
	CSDI	1.67e-02 ± 1.23e-03	2.13e-02 ± 7.50e-04	3.24e-02 ± 8.57e-04
	SADI	**8.62e-03 ± 1.12e-03**	**1.44e-02 ± 5.96e-04**	**2.67e-02 ± 8.21e-04**
NACSE	MICE	2.44e-02 ± 7.06e-05	3.68e-02 ± 6.75e-05	5.14e-02 ± 6.48e-05
	BRITS	2.47e-02 ± 7.77e-05	3.13e-02 ± 1.00e-04	4.09e-02 ± 3.52e-04
	SAITS	2.22e-02 ± 4.45e-05	3.02e-02 ± 8.25e-05	4.67e-02 ± 6.09e-05
	CSDI	7.82e-02 ± 4.36e-04	7.60e-02 ± 2.45e-04	7.44e-02 ± 1.89e-04
	SADI	**1.41e-02 ± 6.50e-05**	**1.48e-02 ± 3.45e-05**	**1.85e-02 ± 4.48e-05**
Air Quality	MICE	2.91e-02 ± 4.28e-04	2.80e-02 ± 1.17e-04	2.81e-02 ± 5.57e-05
	BRITS	2.88e-02 ± 3.84e-04	2.81e-02 ± 5.88e-05	2.88e-02 ± 3.65e-04
	SAITS	2.88e-02 ± 4.28e-04	2.72e-02 ± 1.44e-04	2.60e-02 ± 7.67e-05
	CSDI	8.26e-03 ± 4.36e-05	9.09e-03 ± 8.22e-05	1.07e-02 ± 1.06e-05
	SADI	**4.85e-03 ± 1.84e-04**	**5.92e-03 ± 2.38e-06**	**7.65e-03 ± 5.60e-05**

Table 1: Comparing the performance of SADI with other models on random missing scenarios with different percentages of data missingness. The metric used is RMSE ± 95% Confidence Interval. The best performances are bolded.

Dataset	Model	Random missing (%)		
		20%	50%	80%
AgAID	SADI	**8.62e-03 ± 1.12e-03**	**1.44e-02 ± 5.96e-04**	**2.67e-02 ± 8.21e-04**
	SADI (no FDE)	1.44e-02 ± 8.08e-04	1.91e-02 ± 5.33e-04	3.14e-02 ± 9.92e-04
	SADI (no 2nd block)	1.82e-02 ± 1.20e-03	3.25e-02 ± 1.38e-03	5.04e-02 ± 1.07e-03
	SADI (no weighted comb)	4.68e-02 ± 1.95e-03	4.64e-02 ± 1.72e-03	4.59e-02 ± 1.90e-03
NACSE	SADI	**1.41e-02 ± 6.50e-05**	**1.48e-02 ± 3.45e-05**	**1.85e-02 ± 4.48e-05**
	SADI (no FDE)	1.31e-00 ± 2.31e-03	1.11e-00 ± 1.94e-03	8.97e-01 ± 1.40e-03
	SADI (no 2nd block)	6.94e-02 ± 8.12e-04	7.08e-02 ± 3.14e-04	6.37e-02 ± 2.12e-04
	SADI (no weighted comb)	2.17e-01 ± 2.87e-04	2.11e-01 ± 2.32e-04	2.01e-01 ± 1.89e-04

Table 2: Ablation study for different components of SADI. The metric used is RMSE ± 95% confidence interval. The best performances are bolded.

the hourly PM2.5 measurements for 12 months from 36 stations (features) located in Beijing. We aggregate the measurements into 36 time steps for our time-series data following (Song et al. 2020; Cao et al. 2018; Tashiro et al. 2021). The dataset has 13% original missing data. It contains a distinct test set to evaluate the model.

Experimental Setup

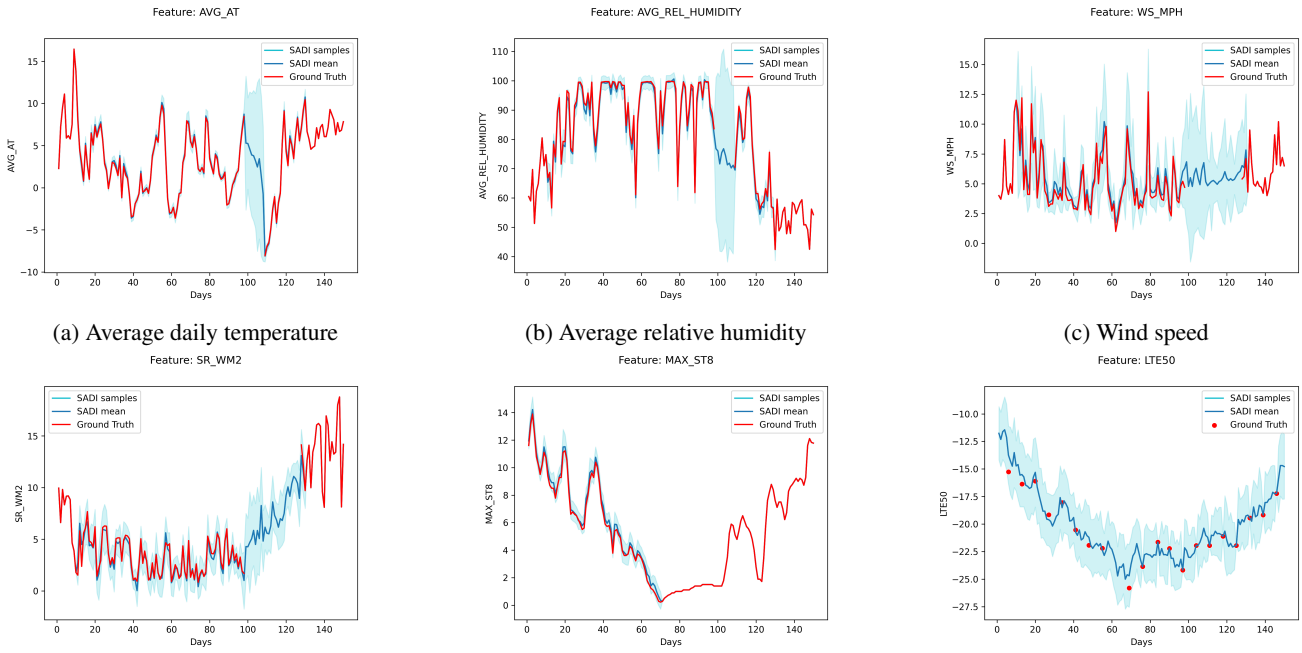
We evaluated our model, SADI, with the metric root mean square error (RMSE) along with 95% confidence interval. We compared our model with four other models such as - MICE (van Buuren and Groothuis-Oudshoorn 2011), an iterative linear regression-based model; BRITS (Cao et al. 2018), a bidirectional RNN-based autoregressive model; SAITS (Du, Côté, and Liu 2023), a self-attention mechanism-based model; and CSDI (Tashiro et al. 2021), another conditional diffusion-based model. We evaluated the models on random missing scenarios varying the percentage (20%, 50%, 80%) of missing data.

For all the datasets, SADI achieved the lowest Root Mean Square Error (RMSE) in all scenarios of missing data percentages, as illustrated in Table 1. For the NACSE dataset, which has the largest data dimensions in our study, we had to adjust one of the CSDI hyperparameters, specifically the number of channels used for data processing from 64, which is recommended, to 8 to fit within the GPU memory limits. This adjustment significantly impaired the performance of CSDI on the NACSE dataset, as can be seen in Table 1. In contrast, SADI does not encounter this issue. It scales efficiently with the data size and consumes fewer GPU resources during runtime, ensuring stable and superior performance without compromising on the model's effectiveness.

Ablation Study

We did an ablation study with respect to the three features of SADI: (1) the FDE (feature dependency encoder) block that models feature inter-correlations, (2) the two-stage imputation process, and (3) the weighted combination of the two intermediate noise predictions. The models evaluated in the ablation study are as follows:

1. **SADI**: SADI with all of its components.
2. **No FDE**: SADI without the *feature depepnednecy encoder* (FDE) component.



(a) Average daily temperature (b) Average relative humidity (c) Wind speed

(d) Solar radiation (e) Maximum soil temperature (f) LTE 50

Figure 2: SADI's imputation results plot for some features of the AgAID dataset. The red line is the *ground truth*, the blue line is the *SADI imputation*, and the cyan-colored shaded region is the 3-standard deviation of the generated imputation samples.

3. **No 2nd block**: SADI after removing the second stage of imputation. Instead of having two separate N_{GTA} layers for each block, we now have a single block with $2 \times N_{GTA}$ layers. It takes the first stage's output as the final imputation.
4. **No weighted comb.**: SADI without the weighted combination of two blocks. It takes the prediction of the second stage as the final imputation.

We conducted our ablation study on the AgAID and NACSE datasets. We observe from the results in Table 2 that different datasets have different effects regarding omitting different components of SADI. For the AgAID dataset, we observe that the *No FDE* model performs slightly worse than SADI compared to the other two ablation models. This could indicate that the correlation among features in the AgAID dataset is less significant for accurate imputation, emphasizing the importance of the other two components—the two-stage imputation process and the weighted combination of intermediate predictions. For the NACSE dataset, we observe that the FDE component is particularly important, as its absence resulted in the highest RMSE values, significantly higher than those of the other two ablation models. This indicates a strong correlation among the features in the NACSE dataset, highlighting the necessity of the FDE component for capturing these interdependencies effectively.

We now present some imputation result plots for selected features of the AgAID dataset to illustrate the performance of SADI. The experimental setup involved removing data for six features from the 21 features in the AgAID dataset for 60 consecutive days. The six features are - average atmospheric temperature (Figure 2a), average relative humidity (Figure 2b), wind speed (Figure 2c), solar radiation (Figure 2d, maximum daily soil temperature (Figure 2e), and grape cold hardiness LTE50 (Figure 2f). SADI generates 50 imputation samples, and we take the mean of these predictions as the final imputed values. In Figure 2, the red line represents the ground truth, the blue line represents the final imputation (the mean of the 50 predicted imputations), and the cyan shaded region represents the variance of the samples. This visualization not only demonstrates the accuracy of the imputed values compared to the actual data but also provides an indication of their uncertainty.

The results show that SADI effectively captures the underlying patterns in the data and provides accurate imputation even when significant portions of the data are missing for contiguous days. The low variance in the cyan-shaded regions indicates a high level of confidence in the imputations. Overall, these plots highlight SADI's robustness and reliability in handling missing data in time-series datasets.

5 Conclusion

In this study, we introduced SADI, a **S**elf-**A**ttention-based **D**iffusion model for time-series **I**mputation, designed to address the challenge of missing data in time-series datasets. SADI explicitly models feature and temporal correlations with the FDE and GTA components and includes a two-stage imputation process that enhances the quality of imputation.

Through extensive experimentation on agricultural, climate, and air quality datasets, we demonstrated that SADI outperformed existing state-of-the-art imputation models in

terms of accuracy and computational efficiency. SADI consistently achieved the lowest root mean square error (RMSE) across all tested datasets and missing data scenarios (20%, 50%, 80%). This robust performance underscores the effectiveness of our model in handling diverse types of time-series data with varying degrees of missingness.

The ablation study highlights the critical roles of the FDE, the two-stage imputation process, and the weighted combination of intermediate predictions. For instance, in the NACSE dataset, feature interdependencies were crucial, showcasing the importance of the FDE component. The two-stage imputation process, which involves an initial rough imputation followed by a refined imputation using a weighted combination of intermediate predictions, significantly enhances the accuracy of the final imputed values.

Unlike the CSDI model, which requires a reduction in the number of channels to fit within GPU memory limits, SADI scales efficiently without compromising performance. This scalability ensures that SADI can handle large-scale datasets and complex models within practical computational constraints. The success of SADI in improving imputation accuracy has significant implications for fields that rely on time-series data, such as agriculture, climate science, and environmental monitoring. Accurate imputation of missing data can lead to better predictive models, more reliable decision-making, and enhanced resource management.

In conclusion, SADI presents a powerful and scalable solution for time-series data imputation, offering substantial improvements over current methods. Its ability to capture complex feature and temporal dependencies ensures high-quality imputation, making it a valuable tool for researchers and practitioners dealing with incomplete time-series data.

Acknowledgements

This research was supported by **NSF and USDA-NIFA under the AI Institute: Agricultural AI for Transforming Workforce and Decision Support (AgAID) award No. 2021-67021-35344**. We thank Lynn Mills and Alan Kawakami for collecting LTE50 data, which we used as a feature in one of the datasets when evaluating the model. We also thank Marcus Keller and his viticulture team at Washington State University for providing their Grape Coldhardiness data. We are grateful to Chris Daly and Dylan Keon for providing us with the NACSE temperature data.

References

Alcaraz, J. M. L.; and Strodthoff, N. 2022. Diffusion-based time series imputation and forecasting with structured state space models. *arXiv preprint arXiv:2208.09399*.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31.

Chen, Y.; Deng, W.; Fang, S.; Li, F.; Yang, N. T.; Zhang, Y.; Rasul, K.; Zhe, S.; Schneider, A.; and Nevmyvaka, Y. 2023. Provably Convergent Schr\" odinger Bridge with Applications to Probabilistic Time Series Imputation. *arXiv preprint arXiv:2305.07247*.

Cini, A.; Marisca, I.; and Alippi, C. 2021. Multivariate Time Series Imputation by Graph Neural Networks. *CoRR*, abs/2108.00298.

Cui, R.; Bucur, I. G.; Groot, P.; and Heskes, T. 2019. A novel Bayesian approach for latent variable modeling from mixed data with missing values. *Statistics and Computing*, 29(5): 977–993.

Du, W.; Côté, D.; and Liu, Y. 2023. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219: 119619.

Fortuin, V.; Baranchuk, D.; Rätsch, G.; and Mandt, S. 2020. Gp-vae: Deep probabilistic time series imputation. In *International conference on artificial intelligence and statistics*, 1651–1661. PMLR.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33: 6840–6851.

Institue, A. 2023. Home : AgAID Institute.

Kong, Z.; Ping, W.; Huang, J.; Zhao, K.; and Catanzaro, B. 2020. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*.

Liu, Y.; Yu, R.; Zheng, S.; Zhan, E.; and Yue, Y. 2019. NAOMI: Non-Autoregressive Multiresolution Sequence Imputation. *CoRR*, abs/1901.10946.

Luo, Y.; Cai, X.; ZHANG, Y.; Xu, J.; and xiaojie, Y. 2018. Multivariate Time Series Imputation with Generative Adversarial Networks. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Luo, Y.; Zhang, Y.; Cai, X.; and Yuan, X. 2019. E²GAN: End-to-End Generative Adversarial Network for Multivariate Time Series Imputation. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, 3094–3100. International Joint Conferences on Artificial Intelligence Organization.

Miao, X.; Wu, Y.; Wang, J.; Gao, Y.; Mao, X.; and Yin, J. 2021. Generative Semi-supervised Learning for Multivariate Time Series Imputation. In *AAAI*.

Oord, A. v. d.; Dieleman, S.; Zen, H.; Simonyan, K.; Vinyals, O.; Graves, A.; Kalchbrenner, N.; Senior, A.; and Kavukcuoglu, K. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

PRISM. 2014. PRISM Climate Group at Oregon State University — prism.oregonstate.edu. https://prism.oregonstate.edu. [Accessed 13-09-2023].

Rasul, K.; Seward, C.; Schuster, I.; and Vollgraf, R. 2021. Autoregressive Denoising Diffusion Models for Multivariate Probabilistic Time Series Forecasting. *CoRR*, abs/2101.12072.

Rubanova, Y.; Chen, R. T. Q.; and Duvenaud, D. K. 2019. Latent Ordinary Differential Equations for Irregularly-Sampled Time Series. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; d'Alché-Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Silva, I.; Moody, G.; Scott, D. J.; Celi, L. A.; and Mark, R. G. 2012. Predicting in-hospital mortality of ICU patients: The PhysioNet/computing in cardiology challenge 2012. *Comput. Cardiol. (2010)*, 39: 245–248.

Sohl-Dickstein, J.; Weiss, E.; Maheswaranathan, N.; and Ganguli, S. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2256–2265. PMLR.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Tashiro, Y.; Song, J.; Song, Y.; and Ermon, S. 2021. CSDI: Conditional Score-based Diffusion Models for Probabilistic Time Series Imputation. *CoRR*, abs/2107.03502.

van Buuren, S.; and Groothuis-Oudshoorn, K. 2011. mice: Multivariate Imputation by Chained Equations in R. *Journal of Statistical Software*, 45(3): 1–67.

Vidotto, D.; Vermunt, J. K.; and Van Deun, K. 2018. Bayesian Latent Class Models for the Multiple Imputation of Categorical Data. *Methodology*, 14(2): 56–68.

Vidotto, D.; Vermunt, J. K.; and Van Deun, K. 2019. Multiple imputation of longitudinal categorical data through bayesian mixture latent Markov models. *Journal of Applied Statistics*, 47(10): 1720–1738.

Yi, X.; Zheng, Y.; Zhang, J.; and Li, T. 2016. ST-MVL: Filling Missing Values in Geo-Sensory Time Series Data. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, 2704–2710. AAAI Press. ISBN 9781577357704.